Method and Apparatus for Evaluating Polynomials and Rational

Functions

Field of the Invention

The present invention relates to a method and apparatus for computing values of

mathematical expressions, and more specifically to a computer processing method and computer

apparatus for computing binary representations of numerical values of polynomials and rational

functions.

Background of the Present Invention

Polynomials and rational functions, which are quotients of polynomials, are used, for

example, by various branches of applied science for determining numerical values of

mathematical expressions for modelling a property of a physical system, for example a rate of

air flow over an airfoil surface. Polynomials can be used to approximate complicated

mathematical expressions. Various methods, for example Horner's Rule that was disclosed in

1819, are used for computing values of polynomials, and provide a degree of accuracy that may

not be acceptable in certain situations, depending on the particular polynomial and the accuracy

required. G.E. Forsythe et al in "Computer Methods for Mathematical Computations"

Prentice-Hall (1977) discloses, in Section 4.2, using Horner's Rule for computing a numerical

value of a polynomial.

Steps for computing binary representations of numbers can create an unacceptably large

deviation between a computed binary representation and its theoretical numerical value due to

successive rounding errors. This can be an intolerable situation when a higher degree of

accuracy is required. Various methods can improve computation accuracy but they may require a significant increase in processing time and/or hardware.

Agarwal et al in "New Scalar and Vector Elementary Functions for the IBM System/370" IBM Journal of Research and Development: Vol.30 No.2 (March 1986), and Gal et al in "An Accurate Elementary Mathematical Library for the IEEE Floating Point Standard" ACM Transactions on Mathematical Software Vol.17 No.1 (March 1991) pp. 26 to 45, appear to disclose methods that include a lookup table having non-equidistantly spaced entries for computing values for elementary mathematical functions.

Markstein in "Computation of Elementary Functions on the IBM RISC System/6000 Processor" IBM Journal of Research and Development Vol.34 No.1 (January 1990) appears to disclose a method for computing values of a function $g(x)$ near a given point x_i.

Gustavson et al in "Elementary Function Subroutines" Department of Mathematical Sciences of IBM Watson Research Center (January 1985) appears to disclose a program implementation for a software mathematical library.

Object of the Present Invention

An object of the present invention is to obviate the disadvantages of the prior art.

Summary of the Present Invention

A first aspect of the present invention provides a machine-processing method for computing a property of a mathematically modelled physical system, the steps including: reading, via a machine processing unit, input data including a value for each identified ordered coefficient of a first polynomial $p(x)$ representing the property, the polynomial $p(x)$ being

expressed as $p(x) = \sum(P_j \cdot x^j)$ where $j = 0$ to n, a value of a quantity x, a value of a predetermined x_i, and a value of a predetermined $p(x_i)$ correspondingly paired with said predetermined x_i; building, via the machine processing unit, a value of a second polynomial $c(x)$ having ordered coefficients, the second polynomial $c(x)$ being expressible as:

5   $c(x) = \sum(C_k \cdot x^k)$ where $k = 0$ to $(n-1)$ so that the first polynomial $p(x)$ is expressible as: $p(x) = p(x_i) + \{x - x_i\} \cdot c(x)$, including the steps of: determining, via the machine processing unit, a value for each ordered coefficient of the second polynomial $c(x)$ by generating a plurality of machine processing unit signals to determine each of the ordered coefficients of the second polynomial $c(x)$ from: $C_k = \sum(P_{(k+1+j)} \cdot x_i^j)$ where $j = 0$ to $(n-1-k)$; determining, via the

10   machine processing unit, a value of the second polynomial $c(x)$ by generating a plurality of machine processing unit signals to determine: $c(x) = \sum(C_k \cdot x^k)$ where $k = 0$ to $(n-1)$; and constructing, via the machine processing unit, a value of the first polynomial $p(x)$ by generating a plurality of machine processing unit signals to determine: $p(x) = p(x_i) + \{x - x_i\} \cdot c(x)$; and outputting, via the machine-processing unit, the value of the first polynomial $p(x)$ representing

15   said property of the mathematically modelled physical system.

Preferably the machine-implementable method is further adapted so that a difference between x and x_i is sufficiently small to achieve a desired accuracy of a final computation of the machine representation of a numerical value of the first polynomial $p(x)$.

Preferably the machine-implementable method is further adapted so that the step of

20   reading the input data includes reading, via the machine processing unit, the input data from a machine-readable medium.

Preferably the machine-implementable method is further adapted so that the ordered coefficients of the second polynomial *c(x)* are computed from a mathematical expression derivable from: $C_k = \sum(P_{(k+1+j)} \cdot x_i^j)$ where $j = 0$ to $(n - 1 - k)$.

Preferably the machine-implementable method is further adapted so that the mathematical expression is a mathematical recurrence expression.

Preferably the machine-implementable method is further adapted so that the mathematical recurrence expression is a forward mathematical recurrence expression.

Preferably the machine-implementable method is further adapted so that the mathematical recurrence expression is a backward mathematical recurrence expression.

Preferably the machine-implementable method is further adapted to implement the backward mathematical recurrence expression by including further steps for: equating, via the machine-processing unit, a value of a highest ordered coefficient of the second polynomial *c(x)* to a value of an identified highest ordered coefficient of the first polynomial *p(x)* by generating a plurality of machine processing unit signals to determine: $C_{n-1} = P_n$; and computing, via a machine processing unit, a value for each lower ordered coefficient of the second polynomial *c(x)* by generating a plurality of machine processing unit signals to determine: $C_{k-1} = x_i \cdot C_k + P_k$ where $k = (n - 1)$ to 1 .

Preferably the machine-implementable method is further adapted so that the predetermined x_i is selected from a set of predetermined values of x_i.

Preferably the machine-implementable method is further adapted so that the predetermined x_i is a closest member of the set to the identified x.

Preferably the machine-implementable method is further adapted so that the step of determining a value of the second polynomial $c(x)$ is computed by using Horner's Rule.

Preferably the machine-implementable method is further adapted for determining a value of a denominator polynomial $q(x)$ having identified ordered denominator coefficients, the denominator polynomial $q(x)$ being expressible as: $q(x) = \sum(Q_j \cdot x^j)$ where $j = 0$ to m, including further steps of: adapting the input data to further include a value for each identified ordered denominator coefficient of the denominator polynomial $q(x)$, a value of a predetermined $q(x_i)$ correspondingly paired with the predetermined x_i; and determining, via the machine processing unit, a value of another polynomial $d(x)$ having ordered denominator coefficients, the another polynomial $d(x)$ being expressible as: $d(x) = \sum(D_k \cdot x^k)$ where $k = 0$ to $(m-1)$so that the denominator polynomial $q(x)$ is expressible as: $q(x) = q(x_i) + \{x - x_i\} \cdot d(x)$, and a value for the denominator polynomial is resolved.

Preferably the machine-implementable method is further adapted so that the first polynomial $p(x)$ is a numerator polynomial $p(x)$, and $p(x)-p(x_i)$ is computed, and $p(x_i)$ is not read.

Preferably the machine-implementable method is further adapted for determining a value of a rational function $r(x)$ being expressible as a quotient of the numerator polynomial $p(x)$ and the denominator polynomial $q(x)$ expressed as $r(x)= \dfrac{p(x)}{q(x)}$, including further steps of: adapting the input data to further including a value of a predetermined $r(x_i)$ correspondingly paired with the predetermined x_i; constructing, via the machine processing unit, a value of the rational function $r(x)$ by generating a plurality of machine processing unit signals to determine:

$$r(x) = r(x_i) \cdot (1 - \frac{q(x)-q(x_i)}{q(x)}) + \frac{p(x)-p(x_i)}{q(x)} \,.$$

Preferably the machine-implementable method is further adapted so that the rational function is an approximation to a Bessel function.

Preferably the machine-implementable method is further adapted so that the rational function is an approximation to an error function (ERF).

5   Preferably the machine-implementable method is further adapted so that the rational function is an approximation to a complementary error function (ERFC).

Preferably the machine-implementable method is further adapted so that the rational function is an approximation to a log gamma function (LGAMMA).

Preferably the machine-implementable method is further adapted so that all values are machine representations of some numerical value, the machine processing unit is a computer processing unit, and each machine representation is a binary representation operable with the computer processing unit, and the machine-readable medium is a computer-readable medium.

A second aspect of the present invention provides a computer-readable program product having computer executable instructions for instructing a computer, the instructions tangibly embodying the machine-implementable method of the first aspect of the present invention.

A third aspect of the present invention provides a computer-readable mathematical software routine library including computer executable instructions for instructing a computer, the instructions tangibly embodying the machine-implementable method of the first aspect of the present invention.

20   Preferably, the computer-readable mathematical software routine library is further adapted so that the library is operatively associated with a software programming language.

A fourth aspect of the present invention provides a machine for computing a property of a mathematically modelled physical system, the steps including: reading, via a machine processing unit, input data including a value for each identified ordered coefficient of a first polynomial $p(x)$ representing the property, the polynomial $p(x)$ being expressed as

5    $p(x) = \sum (P_j \cdot x^j)$ where $j = 0$ to n, a value of a quantity x, a value of a predetermined x_i, and a value of a predetermined $p(x_i)$ correspondingly paired with the predetermined x_i; building, via the machine processing unit, a value of a second polynomial $c(x)$ having ordered coefficients, the second polynomial $c(x)$ being expressible as: $c(x) = \sum (C_k \cdot x^k)$ where $k = 0$ to $(n-1)$ so that the first polynomial $p(x)$ is expressible as: $p(x) = p(x_i) + \{x - x_i\} \cdot c(x)$, including the steps of:

10   determining, via the machine processing unit, a value for each ordered coefficient of the second polynomial $c(x)$ by generating a plurality of machine processing unit signals to determine each of the ordered coefficients of the second polynomial $c(x)$ from: $C_k = \sum (P_{(k+1+j)} \cdot x_i^j)$ where $j = 0$ to $(n - 1 - k)$; determining, via the machine processing unit, a value of the second polynomial $c(x)$ by generating a plurality of machine processing unit signals

15   to determine: $c(x) = \sum (C_k \cdot x^k)$ where $k = 0$ to $(n - 1)$; constructing, via the machine processing unit, a value of the first polynomial $p(x)$ by generating a plurality of machine processing unit signals to determine: $p(x) = p(x_i) + \{x - x_i\} \cdot c(x)$; and outputting, via the machine-processing unit, the value of the first polynomial $p(x)$ representing the property of the mathematically modelled physical system.

20   Preferably, the machine is further adapted so that a difference between x and x_i is sufficiently small to achieve a desired accuracy of a final computation of the machine representation of a numerical value of the first polynomial $p(x)$.

Preferably, the machine is further adapted so that the means for reading the input data includes means for reading, via the machine processing unit, the input data from a machine-readable medium.

Preferably, the machine is further adapted so that the ordered coefficients of the second polynomial *c(x)* are computed from a mathematical expression derivable from:

$C_k = \sum (P_{(k+1+j)} \cdot x_i^j)$ where $j = 0$ to $(n - 1 - k)$.

Preferably, the machine is further adapted so that the mathematical expression is a mathematical recurrence expression.

Preferably, the machine is further adapted so that the mathematical recurrence expression is a forward mathematical recurrence expression.

Preferably, the machine is further adapted so that the mathematical recurrence expression is a backward mathematical recurrence expression.

Preferably, the machine is further adapted to implement the backward mathematical recurrence expression by further including: means for equating, via the machine processing unit, a value of a highest ordered coefficient of the second polynomial *c(x)* to a value of an identified highest ordered coefficient of the first polynomial *p(x)* by generating a plurality of machine processing unit signals to determine: $C_{n-1} = P_n$; and means for computing, via the machine processing unit, a value for each lower ordered coefficient of the second polynomial *c(x)* by generating a plurality of machine processing unit signals to determine:

$C_{k-1} = x_i \cdot C_k + P_k$ where $k = (n - 1)$ to 1.

Preferably, the machine is further adapted so that the predetermined x_i is selected from a set of predetermined values of x_i.

Preferably, the machine is further adapted so that the predetermined x_i is a closest member of the set to the identified x.

Preferably, the machine is further adapted for determining means for determining a value of the second polynomial $c(x)$ is computed by using Horner's Rule.

Preferably, the machine is further adapted for determining a value of a denominator polynomial $q(x)$ having identified ordered denominator coefficients, the denominator polynomial $q(x)$ being expressible as: $q(x) = \sum(Q_j \cdot x^j)$ where $j = 0$ to m, including further steps of: adapting the input data to further include a value for each identified ordered denominator coefficient of the denominator polynomial $q(x)$, and a value of a predetermined $q(x_i)$ correspondingly paired with the predetermined x_i; and determining, via the machine processing unit, a value of another polynomial $d(x)$ having ordered denominator coefficients, the another polynomial $d(x)$ being expressible as: $d(x) = \sum(D_k \cdot x^k)$ where $k = 0$ to $(m-1)$ so that the denominator polynomial $q(x)$ is expressible as: $q(x) = q(x_i) + \{x - x_i\} \cdot d(x)$, and a value for the denominator polynomial is resolved.

Preferably, the machine is further adapted so that the first polynomial $p(x)$ is a numerator polynomial $p(x)$, and $p(x)$-$p(x_i)$ is computed, and $p(x_i)$ is not read.

Preferably, the machine is further adapted for determining a value of a rational function $r(x)$ being expressible as a quotient of the numerator polynomial $p(x)$ and the denominator polynomial $q(x)$ expressed as $r(x) = \dfrac{p(x)}{q(x)}$, including further steps of: adapting the input data to

further including a value of a predetermined $r(x_i)$ correspondingly paired with the predetermined x_i; and constructing, via the machine processing unit, a value of the rational function $r(x)$ by generating a plurality of machine processing unit signals to determine:

$$r(x) = r(x_i) \cdot (1 - \frac{q(x)-q(x_i)}{q(x)}) + \frac{p(x)-p(x_i)}{q(x)} \ .$$

Preferably, the machine is further adapted so that the rational function is an approximation to a Bessel function.

Preferably, the machine is further adapted so that the rational function is an approximation to an error function (ERF).

Preferably, the machine is further adapted so that the rational function is an approximation to a complementary error function (ERFC).

Preferably, the machine is further adapted so that the rational function is an approximation to a log gamma function (LGAMMA).

Preferably, the machine is further adapted so that all values are machine representations of some numerical value, the machine processing unit is a computer processing unit, each machine representation is a binary representation operable with the computer processing unit, and the machine-readable medium is a computer-readable medium.

A fifth aspect of the present invention provides a machine having a computer-readable program product having computer executable instructions for instructing a computer to embody the machine of the fourth aspect of the present invention.

A sixth aspect of the present invention provides a machine having a computer-readable mathematical software routine library including computer executable instructions for instructing a computer to embody the machine of the fourth aspect of the present invention.

Preferably the computer-readable mathematical software routine library of the sixth aspect of the present invention is further adapted so that the library is operatively associated with a software programming language.

Detailed Description of the Figures of the Present Invention

To illustrate aspects of the present invention, the following figures are used, in which:

Fig. 1 depicts a programming flow chart for computing a value of a polynomial $p(x)$ in accordance with the present invention;

Fig. 2 depicts a programming flow chart for computing a value of a rational function $p(x)/q(x)$ in accordance with the present invention; and

Fig. 3 depicts a programming flow chart for using a subroutine for computing a value of a Bessel function $J_0(x)$ in accordance with the present invention.

Detailed Description of a Preferred Embodiment of the Present Invention

The present invention will be described with reference to an exemplary context of a computer processing method and apparatus for computing a binary representation of a numerical value of a polynomial $p(x)$.

5   For embodiments of the present invention, references to computing or calculating values or numbers will be understood to refer to programming instructions for instructing a computer to compute binary representations of numerical values of a mathematical expression or variable. It is understood that computing machinery can manipulate and transform binary representations of numerical values or numbers.

10   A polynomial $p(x)$ can be mathematically expressed as shown in the following equation:

$$p(x) = \sum P_j x^j \quad \text{where } j = 0 \text{ to } n \qquad \text{in which} \qquad p(x) = P_o + P_1 \cdot x + P_2 \cdot x^2 + \ldots + P_n \cdot x^n \quad \textbf{\textit{Equ. 1}}$$

Ordered coefficients of polynomial $p(x)$ are P_0, ..., P_n. Polynomial $p(x)$ has $n+1$ ordered coefficients and is said to have degree n. A numerical value of polynomial $p(x)$ can be computed

15   by reading each coefficient of polynomial $p(x)$ and an identified x and using steps in Equation 1. Following Equation 1 directly would require a computer processor to perform $\frac{n(n+1)}{2}$ multiplications and n additions. To significantly reduce the processing time, Horner's Rule can be applied to Equation 1, in which polynomial $p(x)$ can be expressed in Equation 1a, as shown below:

20    $$p(x) = P_0 + x(P_1 + x(P_2 + x(\ldots(P_{n-1} + x \cdot P_n)\ldots))) \qquad \textbf{\textit{Equ. 1a}}$$

By following Equation **1a**, a program can perform n multiplications and n additions to compute a number for polynomial $p(x)$ using less processing time than a program that follows Equation **1**.

A polynomial $p(x)$ at $x = x_i$ can be expressed as follows:

5 $p(x_i) = \sum P_j x_i^{j}$ where $j = 0$ to n ***Equ.***

2

Therefore, a difference between Equation **1** and Equation **2** can be expressed as follows:

$p(x) - p(x_i) = \sum P_j (x^j - x_i^j)$ where $j = 0$ to n ***Equ.***

3

10 Expanding Equation **3** can provide the following expression:

$p(x) - p(x_i) = \sum \left\{ P_j (x - x_i) \cdot \sum [x^k \cdot x_i^{(j-1-k)}] \right\}$ where $j = 0$ to n, $k = 0$ to $(j-1)$ ***Equ.***

4

For Equation **4**, in the first sum $j = 0$ to n, and in the second sum $k = 0$ to $(j-1)$. It is appreciated that Equation **4** can be expressed as follows:

15 $p(x) - p(x_i) = (x - x_i) \cdot \sum \left\{ \left(\sum P_{(k+1+j)} \cdot x_i^{j} \right) \cdot x^k \right\}$ where $k = 0$ to $(n-1)$, $j = 0$ to $(n-1-k)$ ***Equ.***

4a

For Equation **4a**, in the first sum $k = 0$ to $(n-1)$, and in the second sum $j = 0$ to $(n-1-k)$. By following Equation **4a**, a value for polynomial $p(x)$ can be computed by reading coefficients of the polynomial $(P_1, P_2, \ldots, P_n)$, and x, x_i, and $p(x_i)$. Equation **4a** can be expressed

20 as a combination of Equation **5** and Equation **6**, as follows:

$$p(x) - p(x_i) = \{x - x_i\} \cdot c(x) = \{x - x_i\} \cdot \sum (C_k \cdot x^k) \quad \text{where } k = 0 \text{ to } (n-1) \qquad \textit{Equ.}$$

5

$$C_k = \sum P_{(k+1+j)} \cdot x_i^j \quad \text{where } j = 0 \text{ to } (n-1-k) \qquad \textit{Equ.}$$

6

5   Equation **5** includes an expression for a new polynomial *c(x)* having coefficients, from $C_0, ..., C_{(n-1)}$, that are not known a priori and therefore need to be determined. Equation **6** can be used for determining values of the coefficients of new polynomial *c(x)* by involving the coefficients of polynomial *p(x)* and an identified x_i. Optionally, other expressions for determining values for each coefficient of new polynomial c(x) can be derived from Equation **6**.

10   It is appreciated that Horner's Rule could be used to compute a value for polynomial *c(x)* after coefficients of polynomial *c(x)* have been computed by using Equation **6** or optionally from another expression that could be derived from Equation **6**. By using Horner's Rule, Equation **5** can be expressed as follows:

$$p(x) - p(x_i) = \{x - x_i\} \cdot \{C_0 + x(C_1 + x(C_2 + x(...(C_{n-2} + x \cdot C_{n-1})...))))\} \qquad \textit{Equ.}$$

15   *5a*

         Processing steps that follow Equation **5a** begin within the innermost set of parentheses and proceed outwards to the outermost brackets. Calculated coefficients of new polynomial *c(x)* can subsequently be used in Equation **5a** for computing a numerical value of polynomial *p(x)*.

         Optionally, unknown coefficients of new polynomial *c(x)* can be computed by applying
20   the principle of mathematical recurrence on Equation **6**. For example, a forward mathematical recurrence can allow computation of coefficients of polynomial *c(x)* by beginning with the

lowest ordered coefficient, C_0, and proceeding to the highest ordered coefficient, $C_{(n-1)}$.

Optionally, a backward mathematical recurrence can compute coefficients of polynomial $c(x)$ by

beginning with the highest ordered coefficient, $C_{(n-1)}$ and proceeding to the lowest ordered

coefficient, C_0. With each step of backwardly computing a coefficient of polynomial $c(x)$, a next

5   outer most bracketed term of Equation **5a** can be advantageously computed.

Optionally, a backward recurrence for determining each coefficient of new polynomial

$c(x)$ can be realized by using Equation **7** and Equation **8**. Each subsequent lower-ordered

unknown coefficient of new polynomial $c(x)$ can be computed by using Equation **8**.

$$C_{n-1} = P_n \qquad\qquad\qquad\qquad\qquad \textbf{\textit{Equ.}}$$

10   **7**

$$C_{k-1} = x_i \cdot C_k + P_k \qquad \text{where } k = (n-1) \text{ to } 1 \qquad\qquad \textbf{\textit{Equ.}}$$

8

To compute a numerical value of a polynomial $p(x)$, a computer can read input data that

includes a value for each identified ordered coefficient of the first polynomial $p(x)$, a value of a

15   quantity x, a value of a predetermined x_i, a value of a predetermined $p(x_i)$ that is correspondingly

paired with said predetermined x_i. An optional step can select a nearest x_i that is closest to an

identified x, in which the nearest x_i is selected from a set of predetermined x_i, and a nearest

predetermined $p(x_i)$ corresponds to the nearest x_i. Optionally, coefficients of polynomial $c(x)$ can

be computed by following Equation **7** and Equation **8**. A value for polynomial $p(x)$ can then be

20   computed by following Equation **5a**. The operations of computing the coefficients of

polynomial $c(x)$ in Equation **8** can be interleaved with computing the nested parenthesis in

Equation **5a** so that each time a new C_k is computed by Equation **8** the next outer nested parenthesis of Equation **5a** is computed next.

Provided $p(x) - p(x_i)$ does not have a multiple root at $x = x_i$, the relative round-off error in computing $p(x) - p(x_i)$ with Equation **5** can be kept small by keeping x sufficiently close to x_i. This can be achieved by having available a number of sufficiently closely spaced x_i's covering the range of x to be handled, and using the nearest x_i to x. To obtain a reasonable result that would be satisfactory for a given computational purpose, a difference between x and x_i can be sufficiently small enough to achieve a desired accuracy of a computed value of polynomial $p(x)$.

Optionally, accuracy can be further improved by choosing x_i such that $p(x_i)$ has extra accuracy beyond the precision of the floating-point number system of the computer. This can be achieved by selecting x_i to be a floating-point number, by exhaustively searching from a starting desired value for x_i, until an x_i is found for which $p(x_i)$ is very close to a floating-point number (i.e. closer than half the distance between adjacent floating-point numbers). Such a floating-point approximation to $p(x_i)$ has an accuracy beyond the precision of the floating-point number system.

Fig. **1** shows the preferred embodiment of the present invention, which is represented as a computer processing method embodied in a programming flow chart for computing a machine representation of a numerical value of a polynomial $p(x)$. Optionally, coefficients of $p(x)$ can be placed in a vector of length n. The method optionally includes a step of accessing a table having a set of predetermined values of x_i and a corresponding set of predetermined values of $p(x_i)$ each paired with a corresponding x_i. A table can contain values for the following expressions:

$$x_i, \quad p(x_i) \qquad \text{for } i = 1 \text{ to } N. \qquad\qquad\qquad Equ.$$

8a

In step **10**, the method of the preferred embodiment begins. In step **15**, each coefficient of polynomial *p(x)* and an identified *x* can be read by a computer processing unit. In step **20**, the computer optionally selects a predetermined x_i from a set of predetermined values of x_i. Preferably, the predetermined x_i is a closest member of the set to the identified *x*. In step **30**, the initial conditions for a programming loop are set. Variable *ksum* is an ongoing summation that represents the sum in Equation **5a**. Variable *ksum* is initially set equal to coefficient P_n of polynomial *p(x)*. Variable *ck* is a temporary value for any coefficients of polynomial *c(x)*, in which variable *ck* is computed to be a specific coefficient of polynomial *c(x)* that is determined in step **50** as will be shown below. Variable *ck* is initially set equal to P_n, which is the highest ordered coefficient of polynomial *p(x)*. Variable *k* is an index counter for a processing loop, in which variable *k* is initially set equal to (n-1), which is one less than the degree of polynomial *p(x)*. In step **40**, a query is performed to determine whether a value for variable *k* is equal to zero. If the numerical value for variable *k* is greater than zero, the method proceeds to step **50**. If the numerical value for variable *k* is equal to zero, the method proceeds to step **60**. In step **50**, a value of variable *ck* is computed, which is coefficient C_k by using the backward recurrence of Equation **8** that involves a higher ordered coefficient of polynomial *p(x)*, which is P_k, and a previously determined coefficient, C_{k-1}, of polynomial *c(x)*. Next, an updated value for variable *ksum* is computed, which is an ongoing summation of the sum in Equation **5a**, by determining a sum for the next outer bracketed term of Equation **5a**. For remaining iterations of step **50**, the summation represented in Equation **5a** progresses from an innermost set of parentheses and

proceeds outwardly for each iterative step in an ongoing manner towards the outermost brackets. An updated value of variable k is computed so that the sum in Equation **5a** can be resolved towards its outermost brackets. In step **60**, the method uses a value for polynomial $c(x)$, which is expressed in Fig. **1** as variable *ksum* after variable k equals zero in step **40**. A value of

5   polynomial $p(x)$ is computed as a rearrangement of Equation **5a**, which is expressed as $p(x) = p(x_i) + (x-x_i)*c(x)$. In step **70**, the method for the preferred embodiment can stop or can be repeated from step **10**.

A second embodiment of the present invention provides a computer processing method for computing a machine representation of a numerical value of a rational function $r(x)$. The

10   rational function $r(x)$ is expressed as a quotient of a numerator polynomial $p(x)$ having a set of n predetermined coefficients from P_0, P_1, ..., P_n, so that $p(x) = \sum P_j \cdot x^j$ where $j = 0$ to n, and a denominator polynomial $q(x)$ having m predetermined coefficients from Q_0, Q_1, ..., Q_m, so that $q(x) = \sum Q_j \cdot x^j$ where $j = 0$ to m. A rational function $r(x)$ can be defined in the following expression:

15   $$r(x) = \frac{p(x)}{q(x)}$$ *Equ. 8b*

From Equation **8b**, it follows that:

$$r(x) = r(x_i) \cdot (1 - \frac{q(x)-q(x_i)}{q(x)}) + \frac{p(x)-p(x_i)}{q(x)}$$ *Equ. 9*

20   The second method can optionally include access to a table having a set of predetermined values of x_i and a corresponding set of predetermined values of $q(x_i)$ each paired

with a corresponding x_i, and a corresponding set of predetermined values of $r(x_i)$ each paired with a corresponding x_i. It is understood that each unique x_i is paired with a correspondingly specific $q(x_i)$, and a correspondingly specific $r(x_i)$. A spacing for table values of x_i can be set such that a resulting absolute value of $(x-x_i)$ is small enough to obtain a desired accuracy. The x_i

5   spacing required for a given desired accuracy is typically estimated with a perturbation analysis (using Taylor series, for example), and verified by numerical testing. The closeness of x to x_i needed to achieve a given accuracy in the computed $r(x)$ depends on the particular rational function $r(x)$.

A table can contain values for the following expressions:

10    $x_i, \quad r(x_i), \quad q(x_i) \qquad \text{for } i = 1 \text{ to } N$ *Equ.*

10

$r(x_i) \quad \text{and} \quad q(x_i) \text{ for } i = 1 \text{ to } N$ *Equ.*

11

Accuracy can be further improved by choosing the x_i's such that the following terms

15   shown in Equation **11** can have extra accuracy beyond the precision of the floating-point number system of a computer. This can be achieved by selecting x_i to be a floating-point number, by exhaustively searching from a starting desired value for x_i, until an x_i is found for which so that both the quantities in Equation **11** are very close to floating-point numbers (closer than half the distance between adjacent floating-point numbers). Such floating-point

20   approximations to $r(x_i)$ and $q(x_i)$ have an accuracy beyond the precision of the floating-point number system.

The second method computes values of a numerator polynomial $p(x)$ and a denominator polynomial $q(x)$ by following Equation 5, as follows:

$$p(x) - p(x_i) = \{x - x_i\} \cdot c(x) \qquad\qquad \text{Equ. 12a}$$

$$q(x) - q(x_i) = \{x - x_i\} \cdot d(x) \qquad\qquad \text{Equ. 12}$$

Polynomial $c(x)$ has $(n-1)$ unknown coefficients and polynomial $d(x)$ has $(m-1)$ unknown coefficients. The second method can determine coefficients of $c(x)$ and $d(x)$ and then determine a value for rational function $r(x)$.

Fig. 2 shows the second embodiment of the present invention, which is represented as a computer processing method embodied in a programming flow chart for computing a computer representation of a numerical value of a rational function $r(x) = p(x)/q(x)$. Optionally, steps for accessing a table having a set of various predetermined values for x_i a set of various predetermined values for $q(x_i)$, and a set of various predetermined values for $r(x_i)$ can be included.

In step 110, the second method begins. In step 115, a computer reads values of each coefficient of polynomial $p(x)$ and each coefficient of polynomial $q(x)$, and an identified x. In step 120, a predetermined x_i from a set of predetermined values of x_i can optionally be selected. Preferably, the predetermined x_i is a closest member of the set to the identified x. These predetermined values can be optionally stored in a table or they can be supplied to the computer in some other means. Optionally, the table includes values for various xi, various q(xi)

associated with the corresponding x_i, and various $r(x_i)$ associated with the corresponding x_i.

Optionally, specific values for the x_i's, $q(x_i)$'s, and $r(x_i)$'s are chosen so that the $q(x_i)$'s and $r(x_i)$'s have extra accuracy beyond machine precision in the manner of the accurate table method that is disclosed by the prior art. It is appreciated that the computed result will be more accurate if the

5   table has extra accuracy. Step **130**, step **140**, and step **150** are similar to step **30**, step **40**, and step **50** of Fig. **1**, in that step **130**, step **140**, and step **150** provide a value for variable *ksum* to be used in step **160** as will be shown next. In step **160**, a value for variable *dp* (i.e., delta p) is computed,which is a computed value for $p(x)-p(x_i)$ for a numerator polynomial $p(x)$. Variable dp will be used in step **205** as will be shown later. Step **170**, step **180**, and step **190** are similar to

10 step **130**, step **140**, and step **150** of Fig. **2**, in that step **170**, step **180**, and step **190** provide a value for variable *ksum* to be used in step **200** as will be shown next. In step **200** a value for variable *dq* (i.e., delta q), which is a computed value for $q(x)-q(x_i)$ for a denominator polynomial $q(x)$ is computed. Variable *dq* will be used in step **202** and **205** as shown later. In step **202** a numerical value for $q(x)$ is computed by rearranging Equation **12**. In step **205** a value for a

15 rational function $r(x)$ is computed by following Equation **9**. In step **210** the second method can stop or can be restarted from step **110**.

A third embodiment of the present invention provides a computer processing method for computing binary representations of a value for a Bessel function. A programming flow chart can be developed by adapting Fig. **2** for computing a value for a Bessel function $J_0(x)$, which is

20 a Bessel function of the first kind of order 0. The third method can be further adapted to compute a value for various Bessel functions, for example J_0, J_1, Y_0, and Y_1, over a range of x (e.g. [0,8]) by first approximating the Bessel function by a piece-wise rational function. Bessel

functions can typically be approximated by rational functions having numerator and denominator polynomials with the same degree (i.e. same number of coefficients).

To compute a value of a Bessel function $J_0(x)$ using the third method, the first task would be to determine a range that an identified x belongs in. The following is a list of steps for determining the proper range to which x belongs: if the identified x is a NAN (i.e., not a legitimate number, such as $\frac{0}{0}$), or +/- INF (i.e., any non-zero number divided by zero), the method does not compute a value for the Bessel function; if the identified x is less than zero, the method makes the sign of the identified x positive; if the identified x is greater than or equal to zero and less than $3.72 * 10^{-9}$, then the value of the Bessel function is 1, assuming the result is returned in IEEE double-precision floating point; if the identified x is greater than or equal to $3.72 * 10^{-9}$ and less than 1, then the method uses a Taylor series to compute a value for the Bessel function; if the identified x is greater than or equal to 1 and less than 1.5, then the method calls a subroutine that incorporates the present invention for determining a value for the Bessel function. The subroutine will be explained in Fig. 3. In each case where the subroutine is called, arguments P, Q, T, and n of the subroutine are read along with values of x and x_i associated with the particular range that x belongs with; if the identified x is greater than or equal to 1.5 and less than 1.9, then the method calls the subroutine for determining a value for the Bessel function; if the identified x is greater than or equal to 1.9 and less than 2, then the method calls the subroutine for determining a value for the Bessel function; if the identified x is greater than or equal to 2 and less than 4, then the method performs steps that are beyond the scope of the present invention. The steps involve accurate evaluation of a rational approximation to $J_0(x)/(x-z_0)$, where z_0 is the first zero of J_0; if the identified x is greater than or equal to 4 and less

than 4.5, then the method calls the subroutine for determining a value for the Bessel function; if the identified x is greater than or equal to 4.5 and less than 5, then the method calls the subroutine for determining a value for the Bessel function; if the identified x is greater than or equal to 5 and less than 6, then the method performs steps that are beyond the scope of the present invention. The steps involve accurate evaluation of a rational approximation to $J_0(x)/(x-z_1)$, where z_1 is the first zero of J_0; if the identified x is greater than or equal to 6 and 6.4, then the method calls the subroutine for determining a value for the Bessel function; if the identified x is grater than 6.4 and less than 7, then the method calls the subroutine for determining a value for the Bessel function; if the identified x is greater than or equal to 7 and less than 7.5, then the method calls the subroutine for determining a value for the Bessel function; if the identified x is greater than or equal to 7.5 and less than 8, then the method calls the subroutine for determining a value for the Bessel function; and if the identified x is greater than or equal to 8 then the method uses an asymptotic approximation for determining a value for the Bessel function, which is beyond the scope of the present invention.

Fig. **3** shows the third embodiment of the present invention, which is represented as a computer processing method embodied in a programming flow chart for computing a machine representation of a numerical value of a Bessel function $J_0(x)$.

In step **360** a subroutine, which can be called **eval_rat**, of the flow chart of the third embodiment begins. The subroutine inputs an argument x, at which the value of a rational function is desired to be computed, and P and Q, which are vectors containing the coefficients of the numerator and denominator polynomials $p(x)$ and $q(x)$, respectively, of the rational function, and T, which is a table containing a set of predetermined x_i's and corresponding $r(x_i)$'s

and $q(x_i)$'s, and n, which is the degree of $p(x)$ and $q(x)$. In step **362** a computer processing unit reads values for each coefficient of numerator polynomial $p(x)$ and denominator polynomial $q(x)$, in which both polynomials have the same number or coefficients, and also reads an identified x. In step **364**, to achieve improved performance, a nearest x_i and other required data are optionally read from a table. In step **366** various variables are initialized for subsequent use in an instruction loop. One software loop is used because the numerator and the denominator polynomials have the same degree (that is, share the same number of coefficients). This loop is similar to a combination of the two loops involving steps **130** to **150** and **170** to **190** in Fig. **2**. In step **368**, a test determines whether the loop has been completed. In step **370**, variables are computed in a manner similar to that combining steps **150** and **190** in Fig. **1**. In step **372**, variables are computed in a manner similar to that combining steps **160** and **200** to **205** shown in Fig. **1**. Step **374** marks the end of the subroutine of the flow chart of the third embodiment.

For non-elementary special functions, for example the error function (ERF), the complementary error function (ERFC), or the Bessel functions,range-reduction properties, such as those available for the elementary functions, are not known. To directly apply prior art table-driven techniques, for example those used for the elementary functions, for these non-elementary special functions, would require a computer processing unit to read coefficients of each of a large number of polynomials, one associated with each table point. This can make the table unacceptably large.

Computing steps that embody the methods of the present invention can be used to determine values for polynomials or rational functions approximating the non-elementary special function. Optionally, coefficients of a polynomial or rational function associated with

each individual table point, x_i, do not have to be stored; instead, these coefficients can be computed using simple formulas from only one (for polynomials) or two (for rational functions) stored function values per table point, resulting in tables of manageable size. In this sense, the present invention provides a benefit similar to that of range reduction formulas, for functions for

5   which range reduction formulas do not exist.

Mathematical software libraries, including those associated with a software programming language or operating system, can be adapted to instruct a general purpose digital computer to embody the present invention. It is appreciated that a software does not necessarily have to belong to a programming language or operating system. There are also stand-alone

10   libraries of mathematical routines, for example, the IBM™ Engineering and Scientific Subroutine Library, ESSL™.

There are many examples that illustrate the practical application of the present invention for mathematically modelling a property of a physical system. C. M. Close in "The Analysis of Linear Circuits", Harcourt, Brace & World (1966), discloses in Chapters 6 and 11 the use of

15   polynomials for determining frequency response characteristics of electrical systems. The same concepts can be applied to non-electrical systems, such as mechanical, hydraulic, acoustical, and thermal systems. R.C. Weyrick in "Fundamentals of Automatic Control", McGraw-Hill Book Company (1975), discloses in Chapters 2 and 6 the use of polynomials for mathematically modelling physical systems for predicting a behavior of a physical system. D. Halliday and R.

20   Resnick in "Physics: Parts 1 and 2", John Wiley & Sons (1978) disclose in Chapters 3 and 4 the use of polynomials for mathematically modelling and predicting spatial co-ordinates of impact of a projectile being released from an aeroplane.

The present invention can be implemented in a computer-readable computer memory program having a medium for storing and carrying the program to a general purpose computer for instructing a computer processing unit to implement the embodiments of the present invention. A computer-memory product can be a floppy disk or compact disk that can be adapted to carry software that tangibly embodies the present invention.

A computer processing method that tangibly embodies the present invention can be used for computing a number for a mathematical model and for comparing the computed number against an observed behaviour of a physical phenomenon.

The concepts of the present invention can be further extended to a variety of other applications that are clearly within the scope of this invention.

Having thus described the present invention with respect to a preferred embodiment as implemented, it will be apparent to those skilled in the art that many modifications and enhancements are possible to the present invention without departing from the basic concepts as described in the preferred embodiment of the present invention. Therefore, what is intended to be protected by way of letters patent should be limited only by the scope of the following claims.